Greg Berg, Kyle Kirtland

Cal Poly Pomona

CS 4200 Dominick Atanasio

May 3, 2020

Strategies:

We looked into many strategies to use for the evaluation function. We ended up using an evaluation function called Improved Score (IS). This evaluation function took the difference between the number of current players moves minus current opponent moves. We chose this one because of its simplicity, its average win rate, which was 64%, and that there were two other heuristics that were very similar to this one the only difference being the ration of 2:1 and 3:1 to the opponents move. Both of these being more aggressive IS evaluation functions. We chose to only implement the IS evaluation because the code we have for it is not working as we expect it to, so any additional changes to the evaluation function would only cause more reasons as to why it’s not working already.

Analysis:

For this project we got everything working about three days before the due date. The only thing we had left was to debug what was going wrong with the evaluation function and the invalid moves on the board. We had many bugs to get rid of some were easy, but there were a few that were hard. One being the evaluation function, as we did not have enough time to finish it and we did not know what was going wrong. What we think was going wrong is it always places its first move in the same location no matter who goes first. We thought this was just the evaluation being aggressive but when the opponent going first it still choses that. Also, the aggressiveness is not working as intended as it sometimes does not pick the best spot for it to be as far as moves it has and moves the opponent has. Another bug was the computer would place the move it decided to make in an invalid place even though it had the places to move stored within its memory. We eventually found out, after a day or more of debugging, that the backtracking was not passing back the right value, what was causing this was we were making a move and then going into the iterative deepening. This caused it to sometimes chose the final move that was to far ahead of the possible moves it could make at its current location. We solved this by removing the initial move before going into iterative deepening.